

config 1	Data@ ext.Adr.	→	Data@ IRAM-Adr.
config 2	Data@ IRAM-Adr.	→ "Ping" →	Data' @ IRAM-Adr.2
config 3	Data' @ IRAM-Adr.2	→ "Pong" →	Data <sup>n</sup> @ IRAM-Adr.3
config n	Data <sup>n</sup> @ IRAM-Adr.n	→	Result@ ext.Adr.2

(Fig. 1b I, 1b II, 1b III) ↔

(Fig. 1c) ↔

(Fig. 1d) ↔

(Fig. 1f1, 1f2) ↔

Fig. 1a



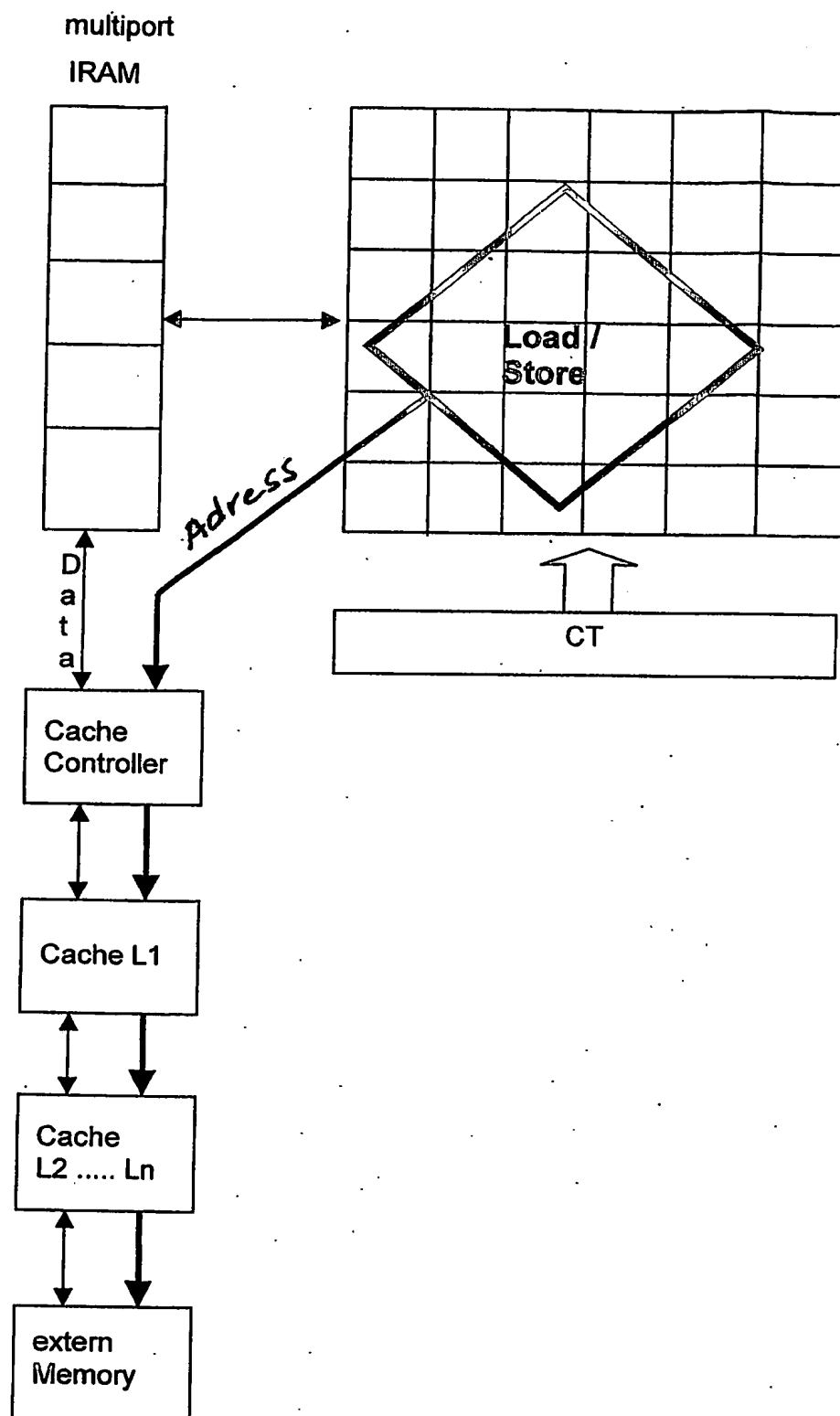


Fig. 1b II

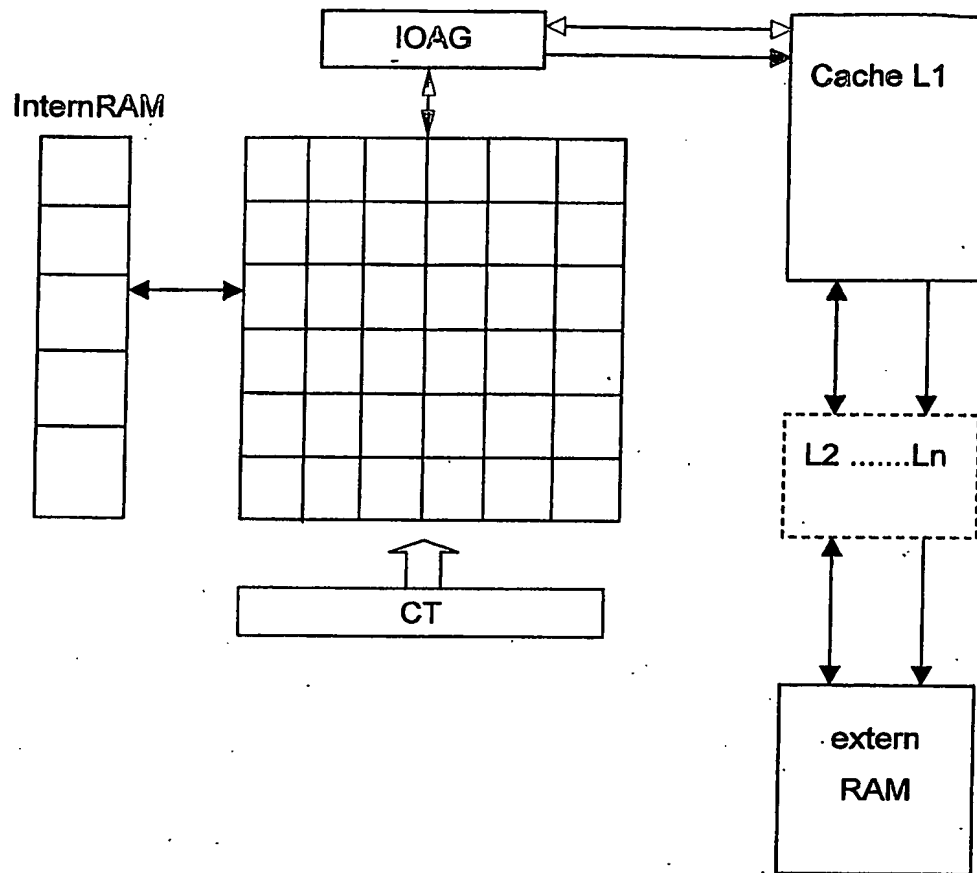


Fig. 1b III

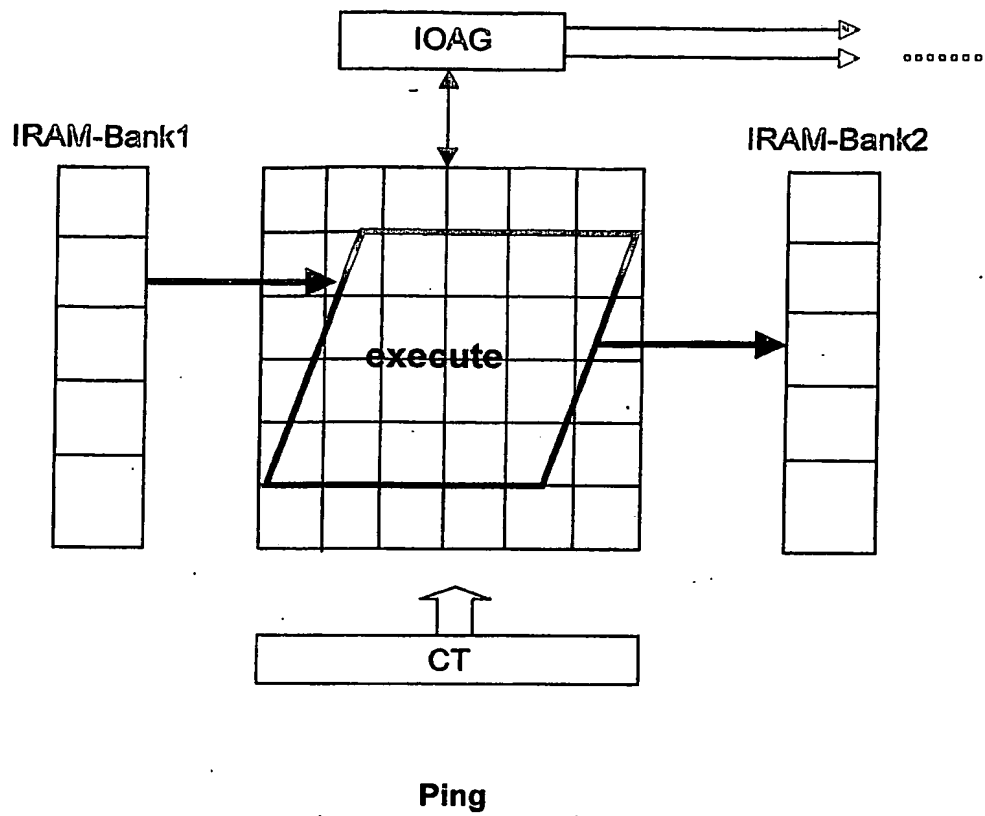


Fig. 1c

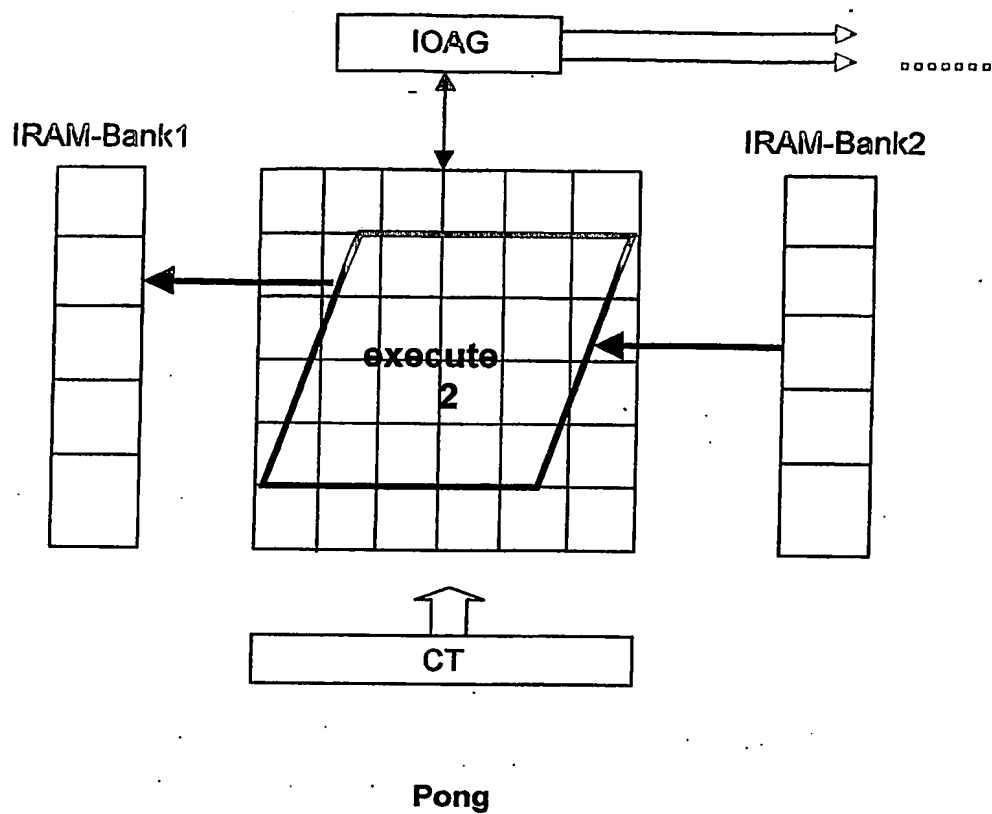
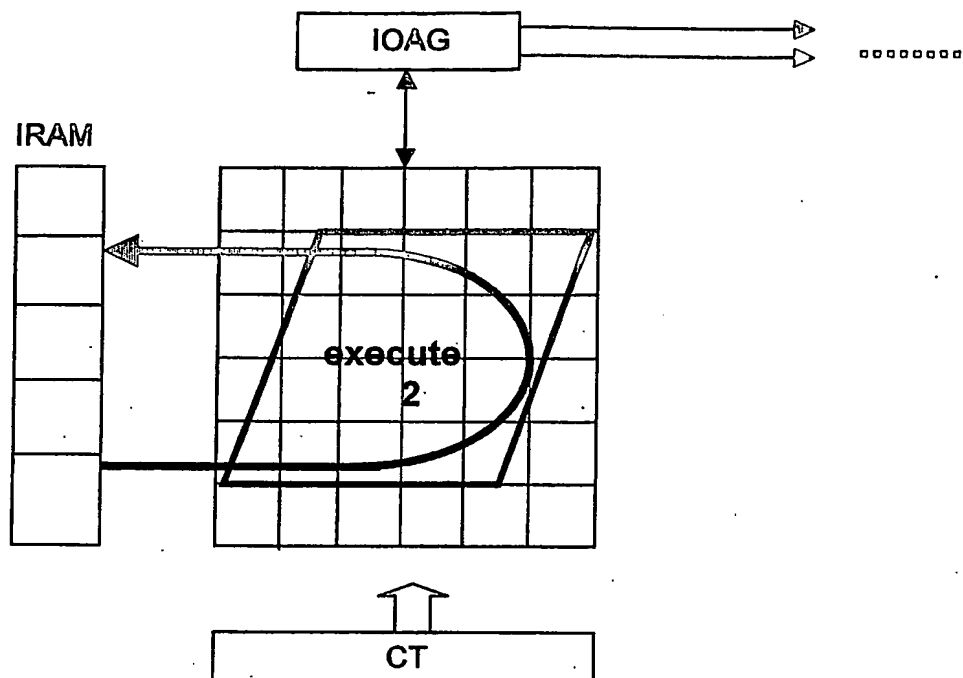


Fig. 1d



Ping / Pong

Fig. 1e1

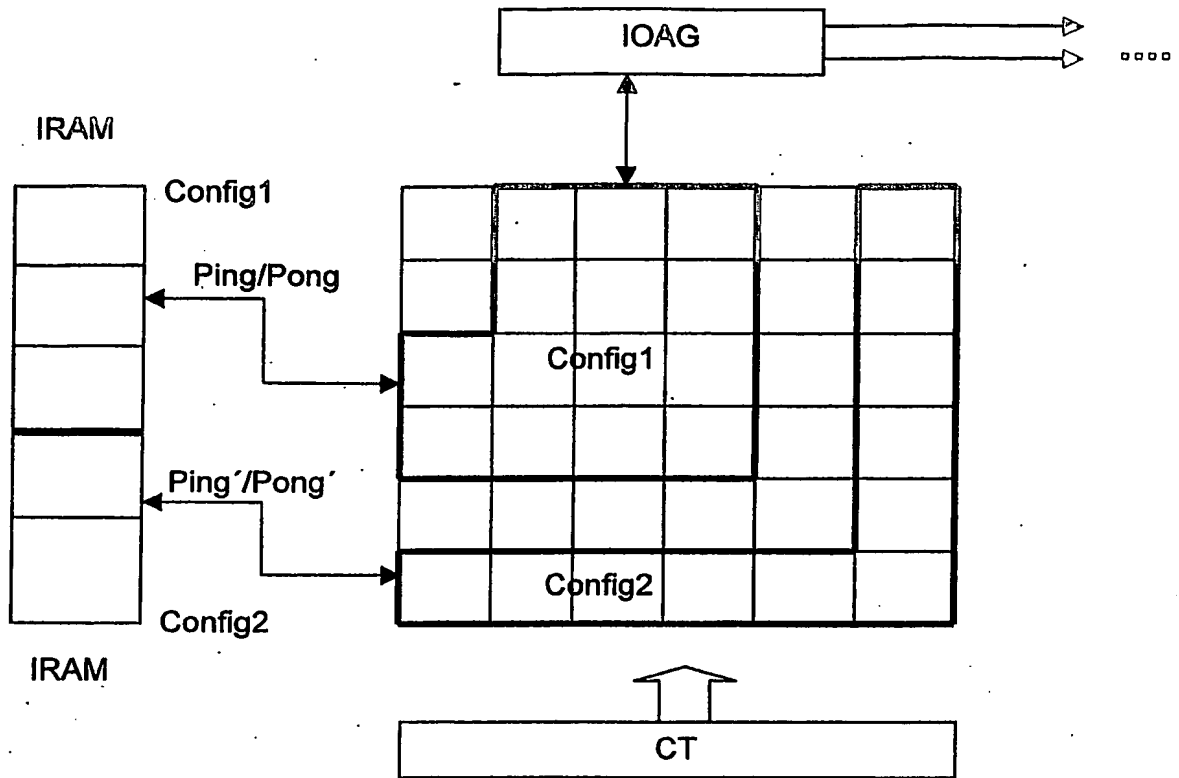


Fig. 1e2



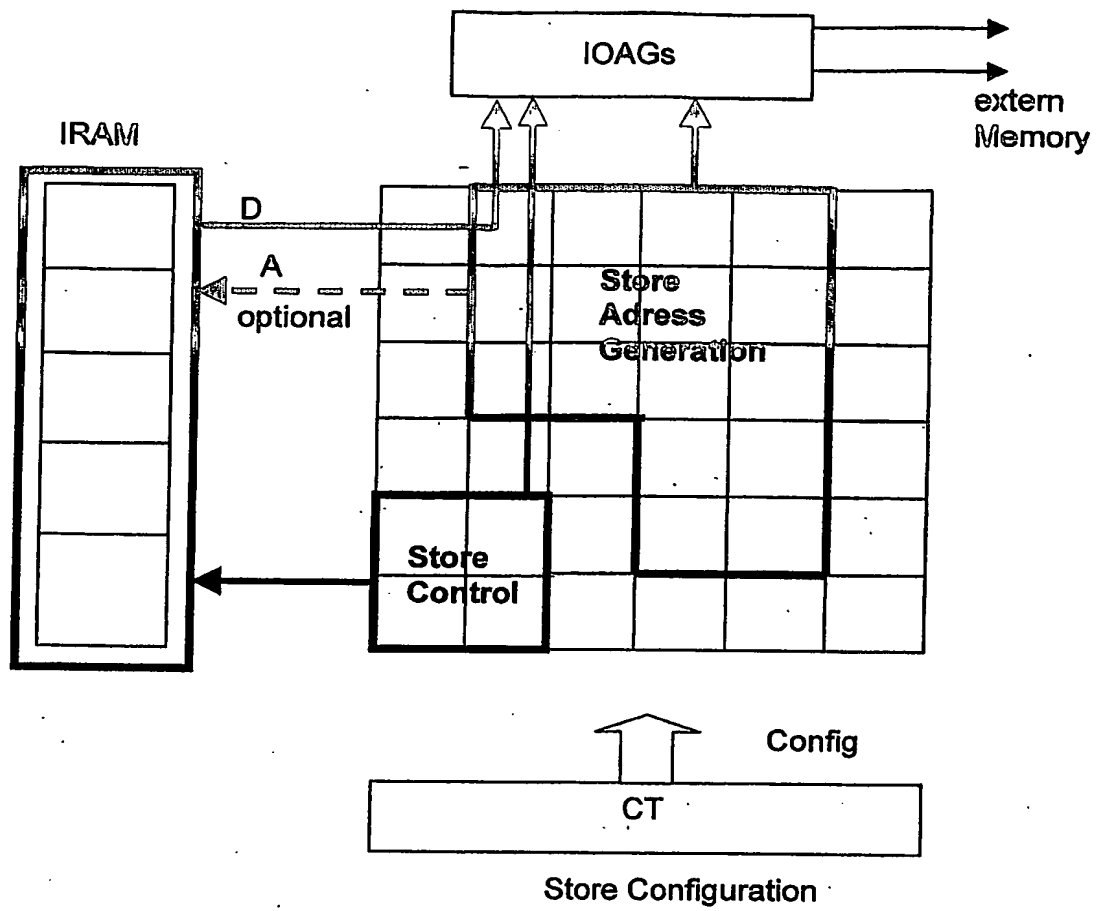


Fig. 1f1

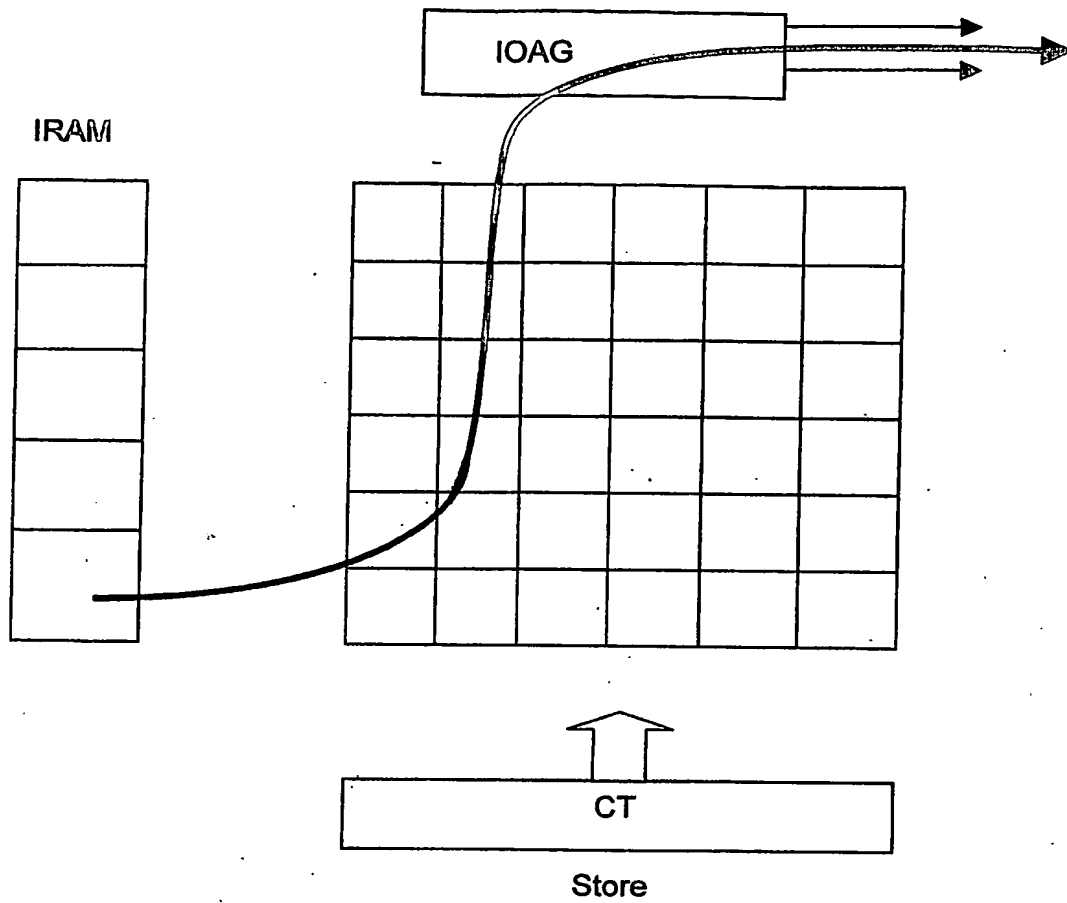


Fig. 1f2

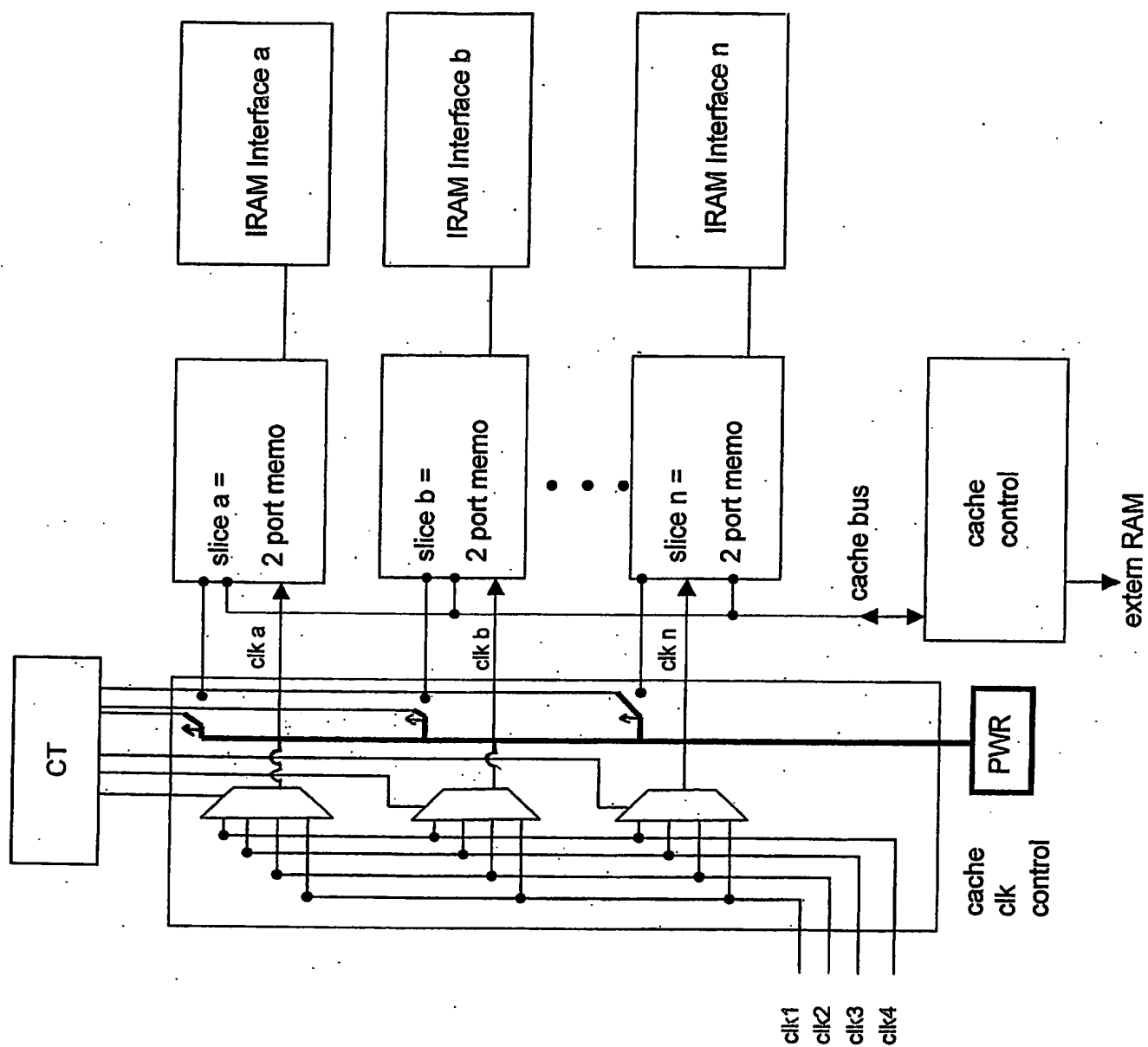


Fig. 2

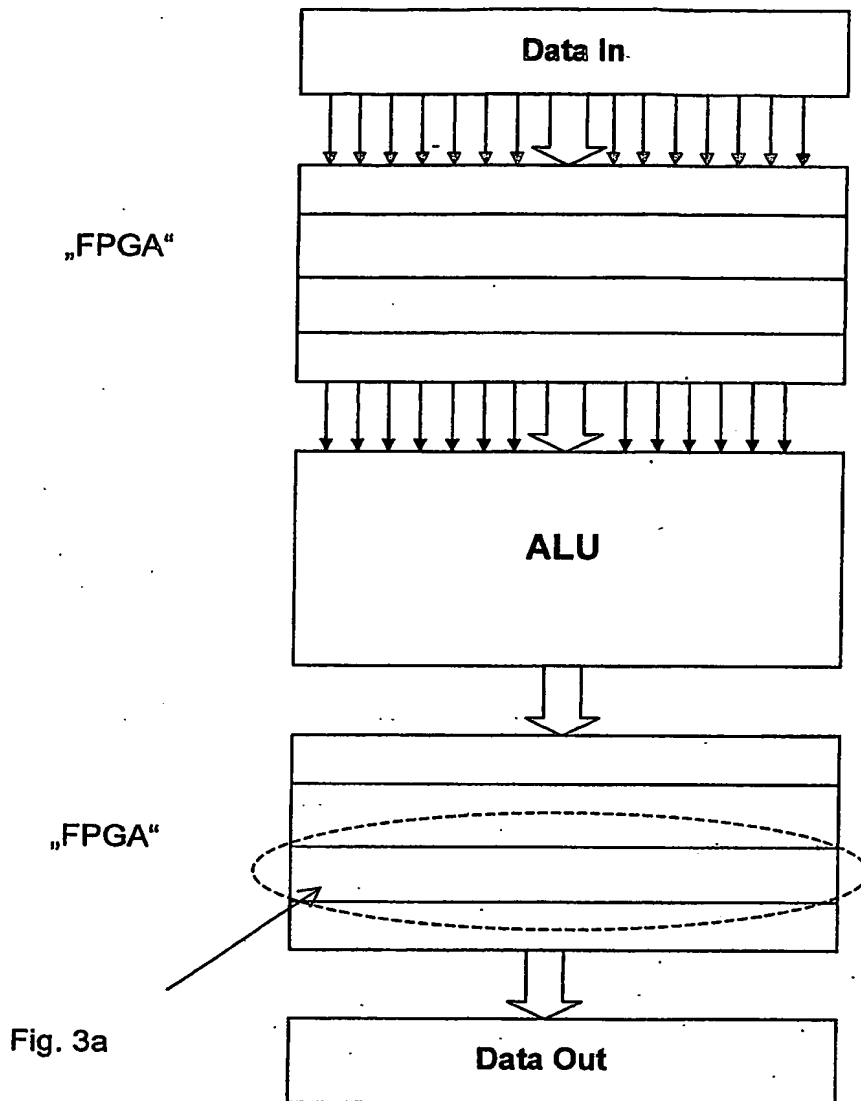


Fig. 3 I

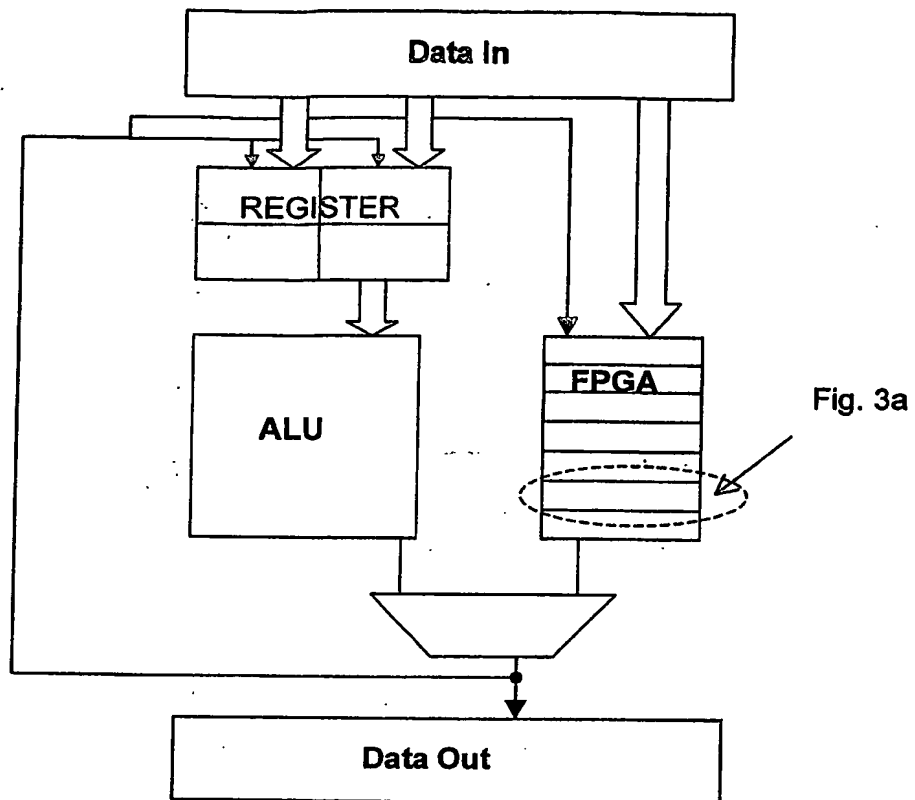


Fig. 3 II

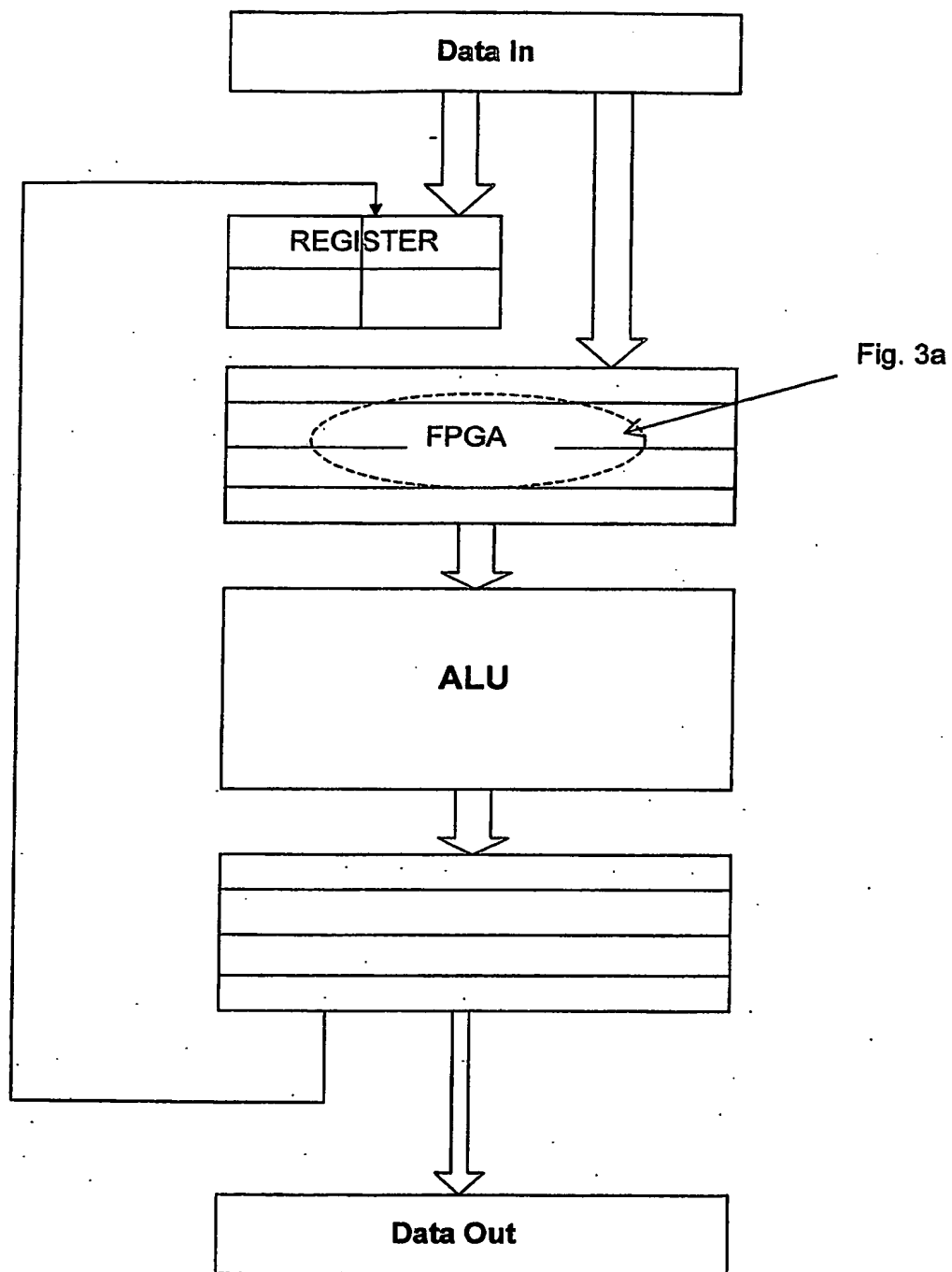


Fig. 3 III

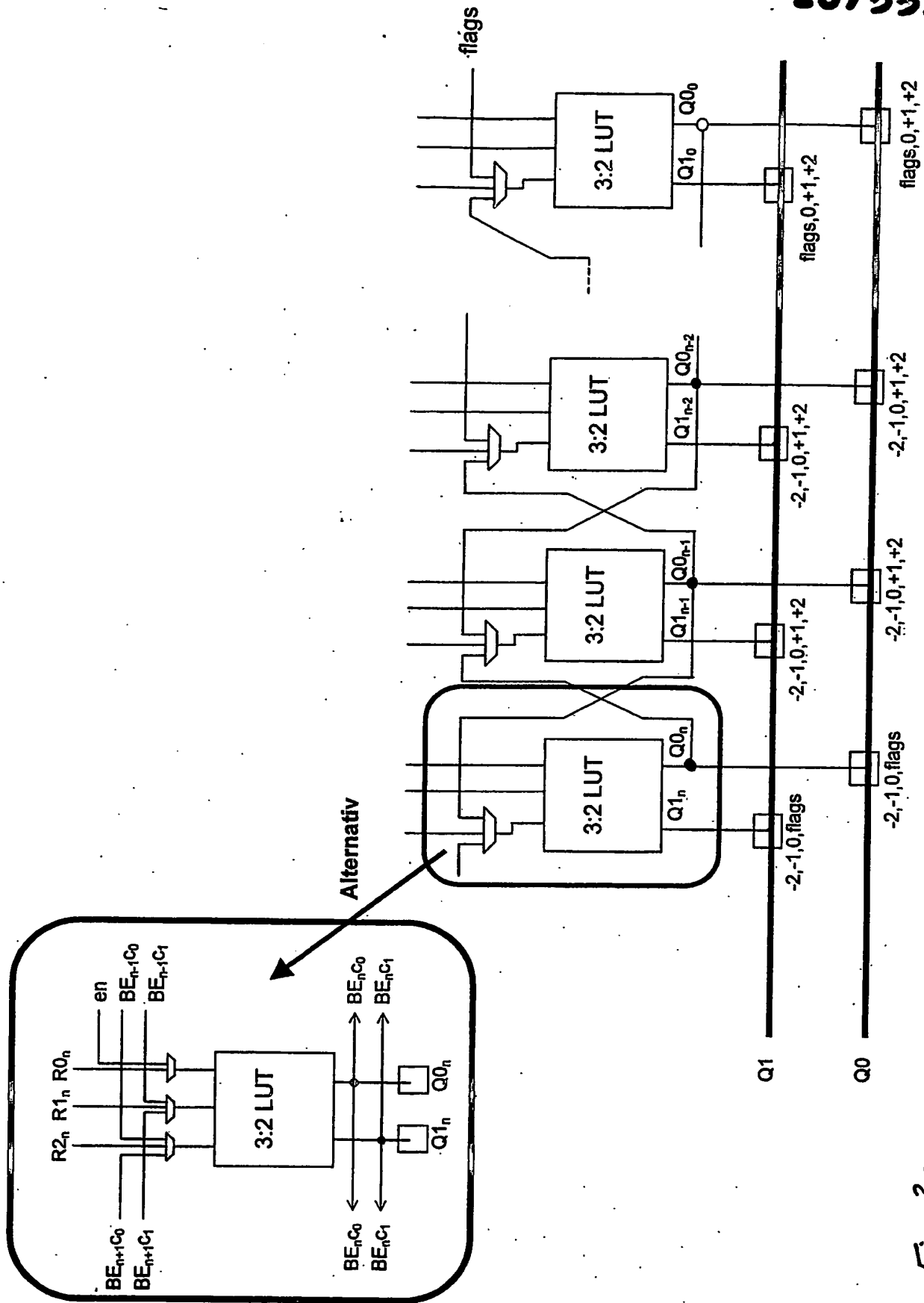
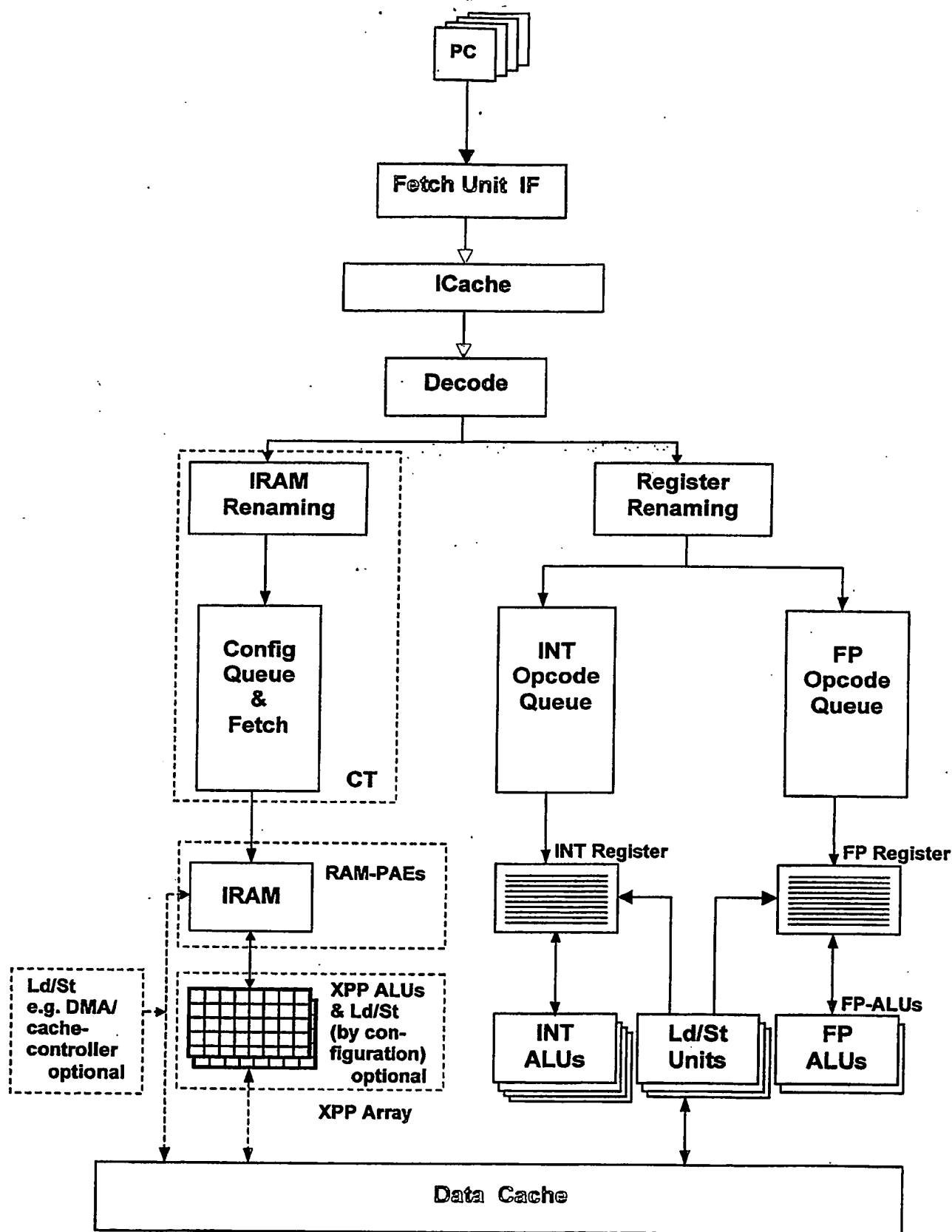


Fig. 3a



Möglicher Aufbau eines SMT-Prozessors mit  
XPP Thread Resource

Fig. 4a



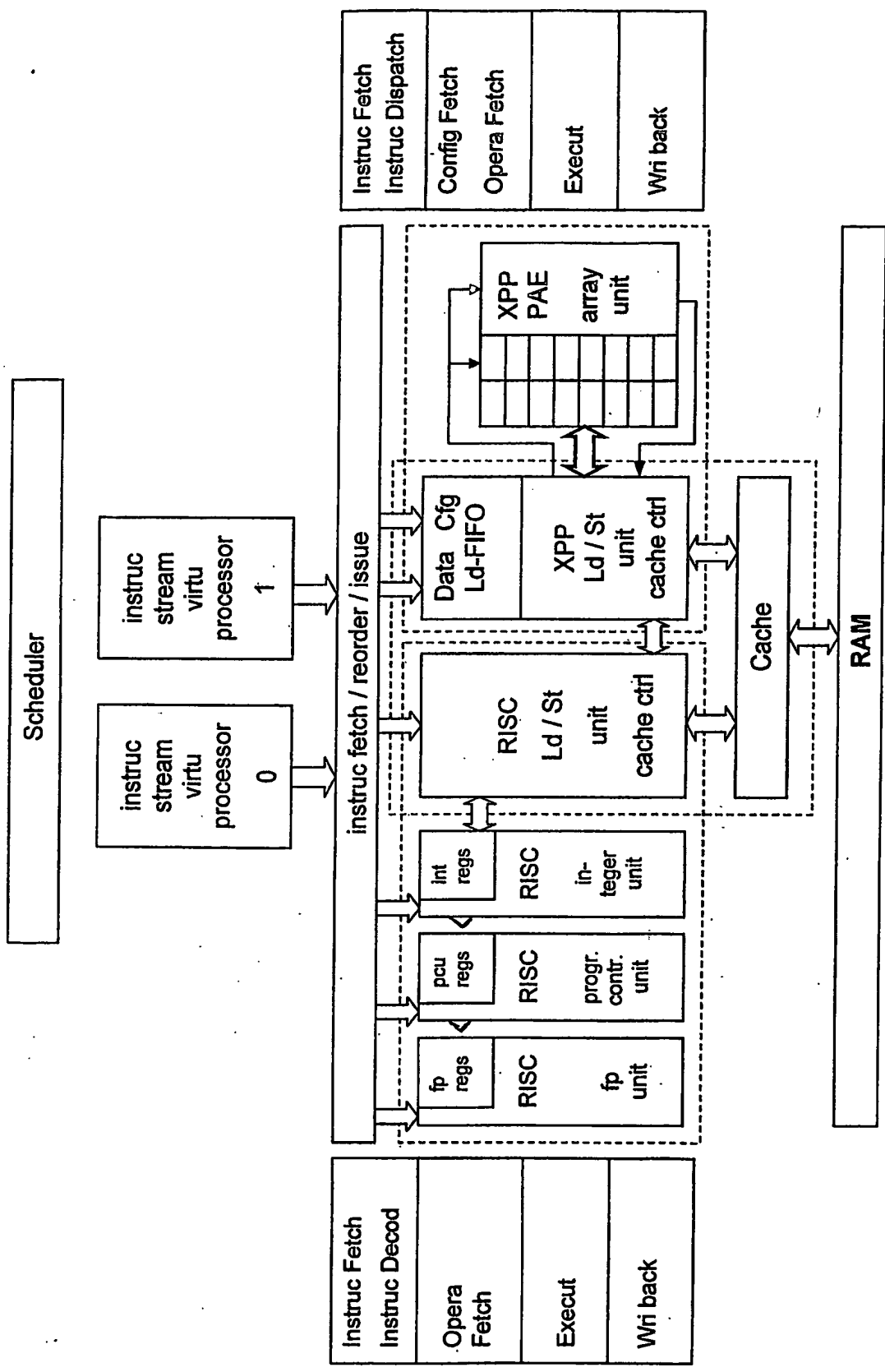


Fig 4b

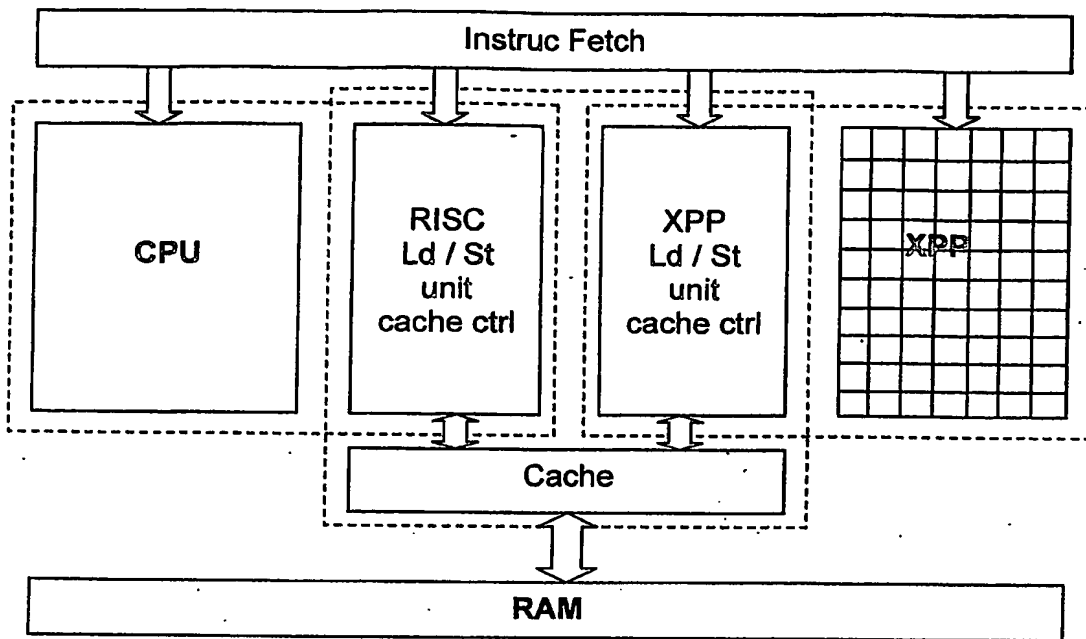


Fig. 4C

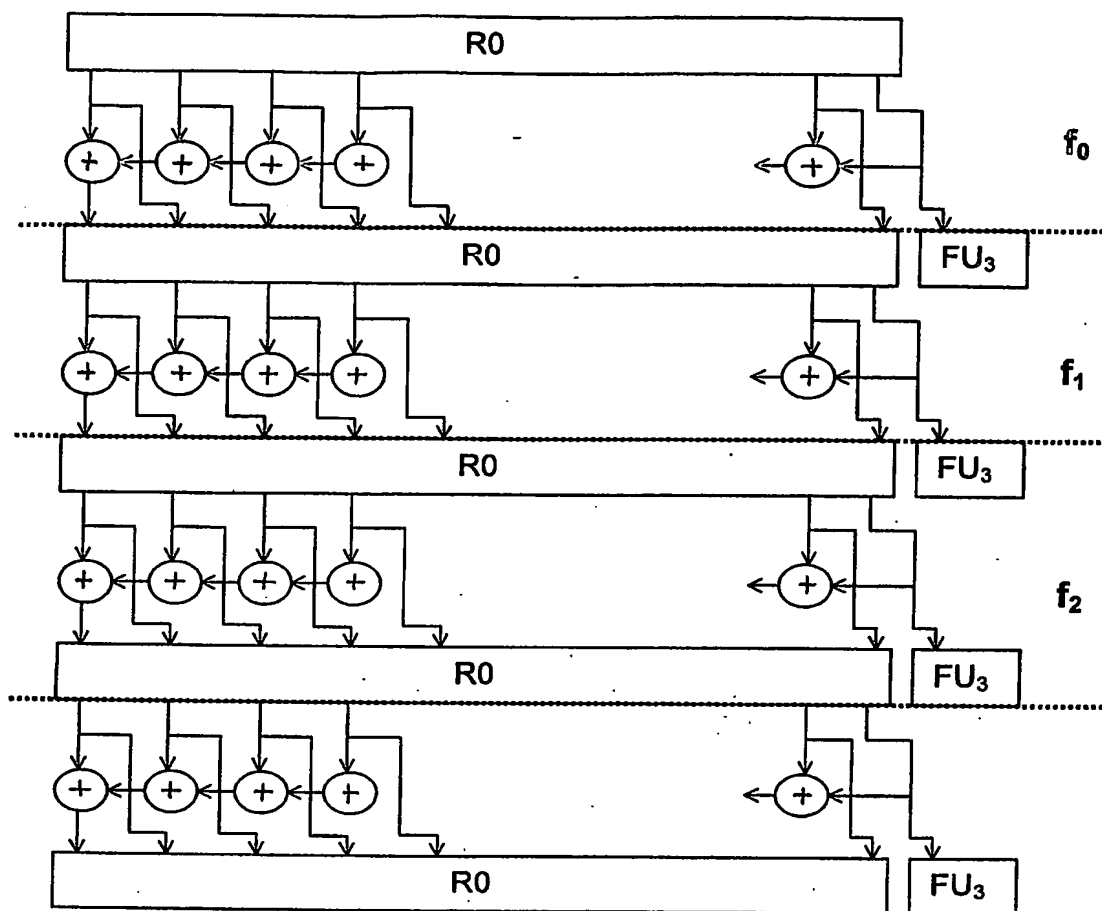
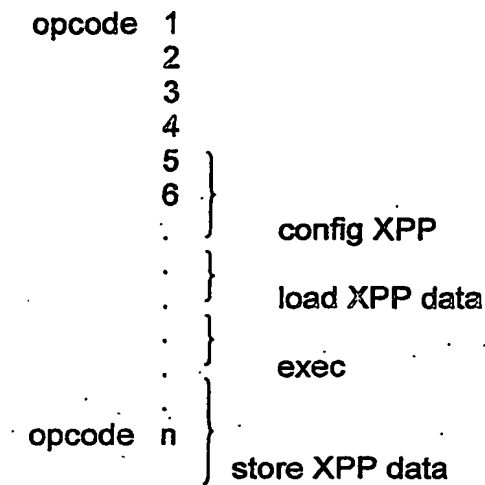


Fig. 5

Scheduler	MEM-Interface	CT Operation	XPP Operation
1 Task 1	NOP	NOP	NOP
2 NOTask	Ld Task 1 Dat	Config Task 1	NOP
3 NOTask	NOP	NOP	Exec Task 1
4 NOTask	Stor Task 1 Dat	NOP	NOP
5 Task 2	Ld Task 2 Dat	Config Task 2	NOP
6 Task 3	Pre-Ld Task 3 Dat	Preconfig Task 3	Exec Task 2
7 Task 4	Pre-Ld Task 4 Dat Stor Task 2 Dat	Preconfig Task 4	Exec Task 3
8 NOTask	Stor Task 3 Dat	NOP	Exec Task 4

Fig. 6a

## XPP im Instruktionsstrom angesteuert durch CPU-Befehle

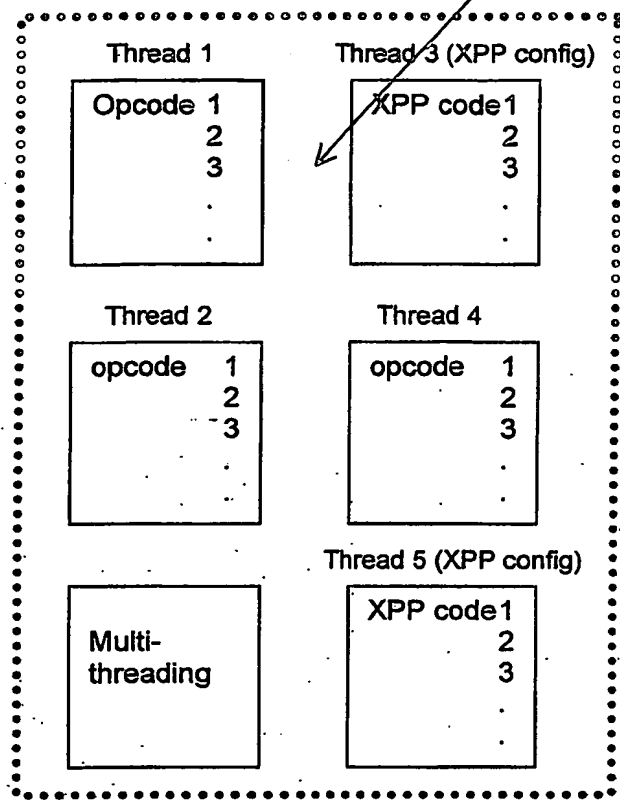


## loosely coupled Coprocessor

## XPP als Thread Resource

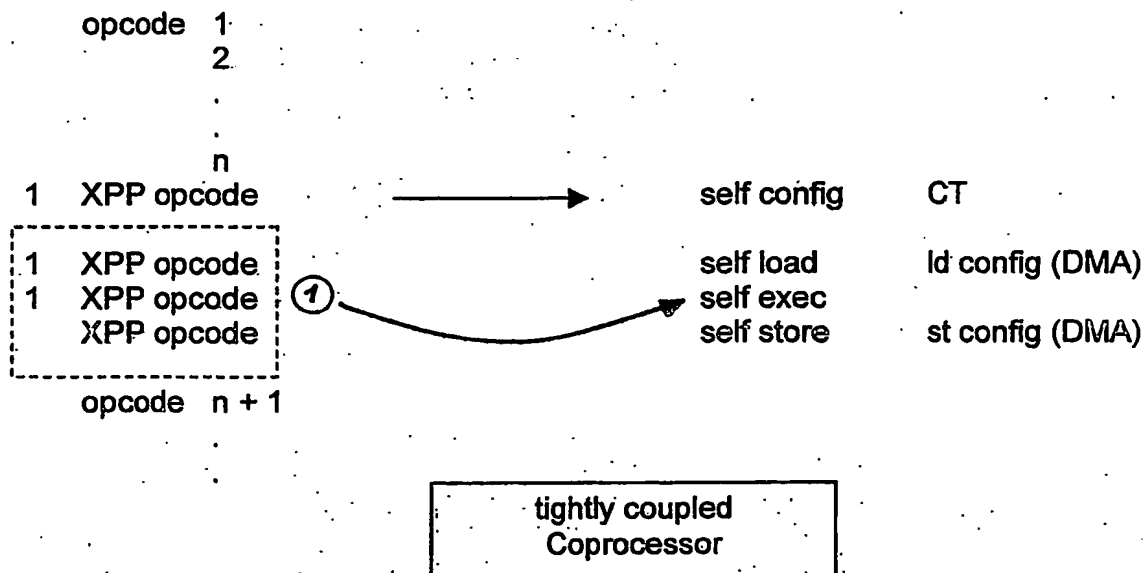
Die Threads sind partitioniert und werden vom Scheduler verwendet

**Fig. 6c**



## XPP im Instruktionsstrom

angesteuert durch XPP-Befehle, die vom IF/ID-Slice separiert werden



**Fig. 6b**